SERIAL NO.: 09/747,056 PETER H. PRIEST (919-806-1600)

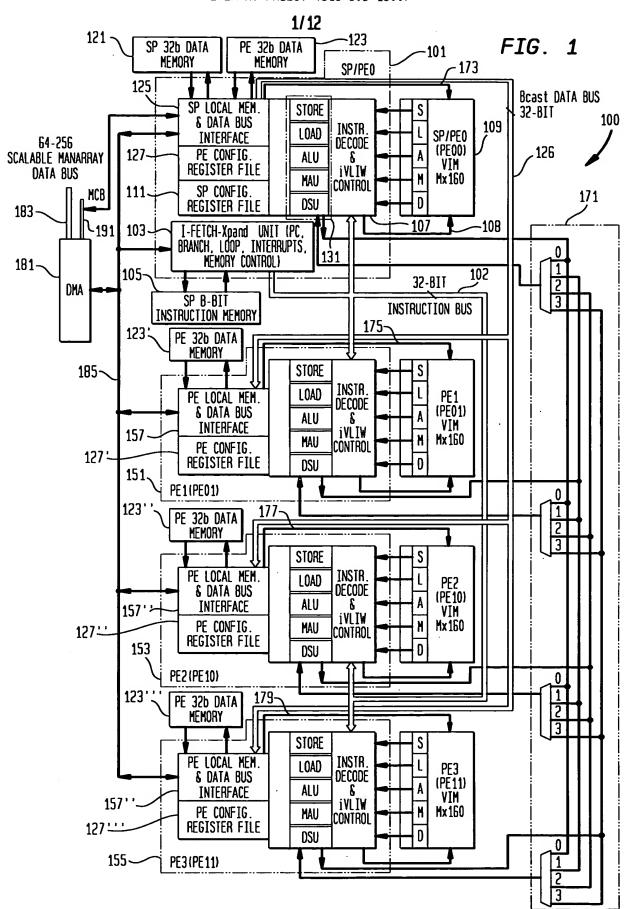
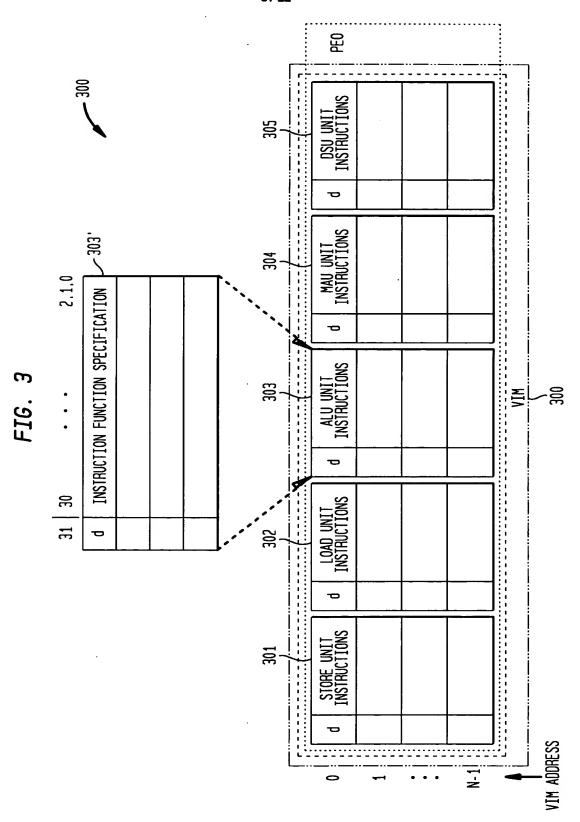


FIG. 2 FETCHED INSTRUCTION **-** 200 INSTRUCTION REGISTER 20~ 24 OUTPUT ENABLES WRITE ENABLES ADDRESS VIM LOAD/STORE 25~ CONTROL WRITE DATA iVLIW MEMORY VIM READ DATA iVLIW REGISTER 26~ TO INSTRUCTION DECODE UNITS

3/12



4/12

5/12

	FIG. 4B	48
XV1- EXECUTE VL IW — 425 Encoding	LIW — 425	430
31 30 29 Group S/P	31 30 29 28 27 26 25 24 23 22 21 20 19 18 Group S/P CtrlOp VX UAF 0 0 0 0 0	17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0 0 0 0 SU LU ALU MAU DSU Vb 0 VimOffs
Syntax/Operation	435 0n	·
Instruction	Operands	Operation
XV1.[SP]	V[01], VIMOFFS, E={SLAMD}, F=[AMDN]	Execute (V[01]+VIMOFFS)[SU] if (E=S)

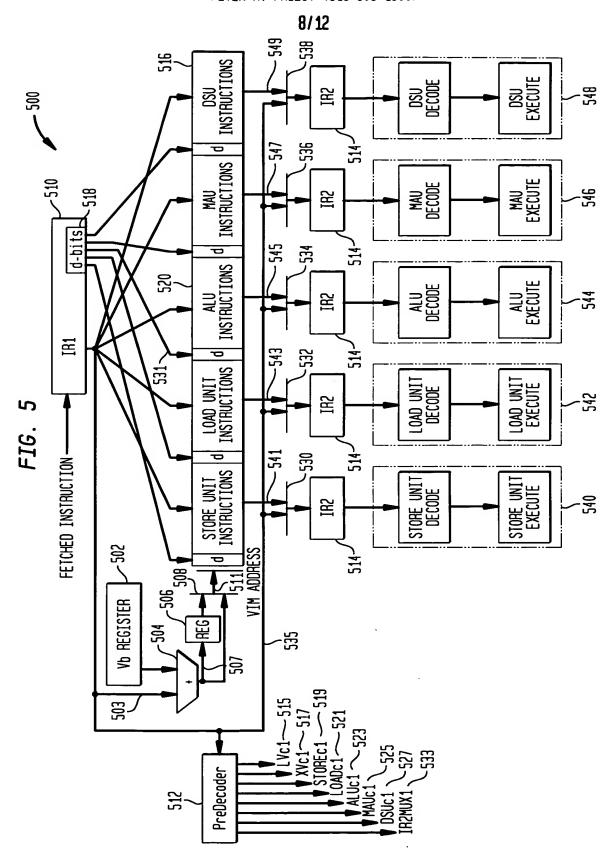
6/12

450 24 23 22 21 20 19 18 17 16 15 14 13 12 11 0 LI d InstrCnt UnitVIM 0 0 0 0 0 LI d InstrCnt UnitVIM 0 0 0 0 0 LI d InstrCnt UnitVIM 0 0 0 0 0 LI d Instruction disable bit 0 0 0 0 0 0 0 0 0 0			
27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11	LV2- LOAD/MOD; Encoding	FY VLIW- 2 — 455	450
CtrlOp 0 LI d InstrCnt Uni Operands LI. u=UnitVIM. V[01]. VIMOFFS. InstrCnt. d		27 26 25 24 23 22 21 20 19 18	14 13 12
Operands LI. u=UnitVIM, V[01], VIMOFFS, InstrCnt, d		0 LI d InstrCnt	0 0 0 0 0 0 0 0 0
Operands LI. u=UnitVIM. V[01]. VIMOFFS. InstrCnt. d	Syntax/Operati		
LI, u=UnitVIM, V[01], VIMOFFS, InstrCnt, d	Instruction	Operands	Operation
LI, u=UnitVIM, V[01], VIMOFFS, InstrCnt, d			<pre>if (LI=0) Load disable bit only disable bit @ (V[01]+VIMOFFS)[UnitVIM] d</pre>
LI, u=UnitVIM, V[01], VIMOFFS, InstrCnt, d			<pre>if (LI=1) Load instructions disable bit @ (V[01]+VIMOFFS)[UnitVIM] d</pre>
LI, u=UnitVIM, V[01]. VIMOFFS, InstrCnt, d			Load next InstrCnt instructions into (V[01]+VIMOFFS)[UnitVIM] —— 1st Instruction following LV2
(V[01]+VIMOFFS)+InstrCnt)[UnitVIM] —— (InstrCnt Instruction following LV2 Instruction following LV2 InstrCnt is a binary coded number, 0 thru F, tha represents from 1 to 16 instructions that can be loaded into up to 16 consecutive UnitVIM location	LV2.[SP]	LI, u=UnitVIM, V[01]. VIMOFFS, InstrCnt, d	(V[01]+VIMOFFS)+1)[UnitVIM]→— 2nd Instruction following LV2
(V[01]+VIMOFFS) +InstrCnt) [UnitVIM] —— (InstrCnt Instruction following LV2 InstrCnt is a binary coded number, 0 thru F, that represents from 1 to 16 instructions that can be loaded into up to 16 consecutive UnitVIM location			• • •
InstrCnt is a binary coded number, 0 thru F, tha represents from 1 to 16 instructions that can be loaded into up to 16 consecutive UnitVIM location			(V[01]+VIMOFFS)+InstrCnt)[UnitVIM] (InstrCnt) th Instruction following LV2
			InstrCnt is a binary coded number, 0 thru F, that represents from 1 to 16 instructions that can be loaded into up to 16 consecutive UnitVIM locations

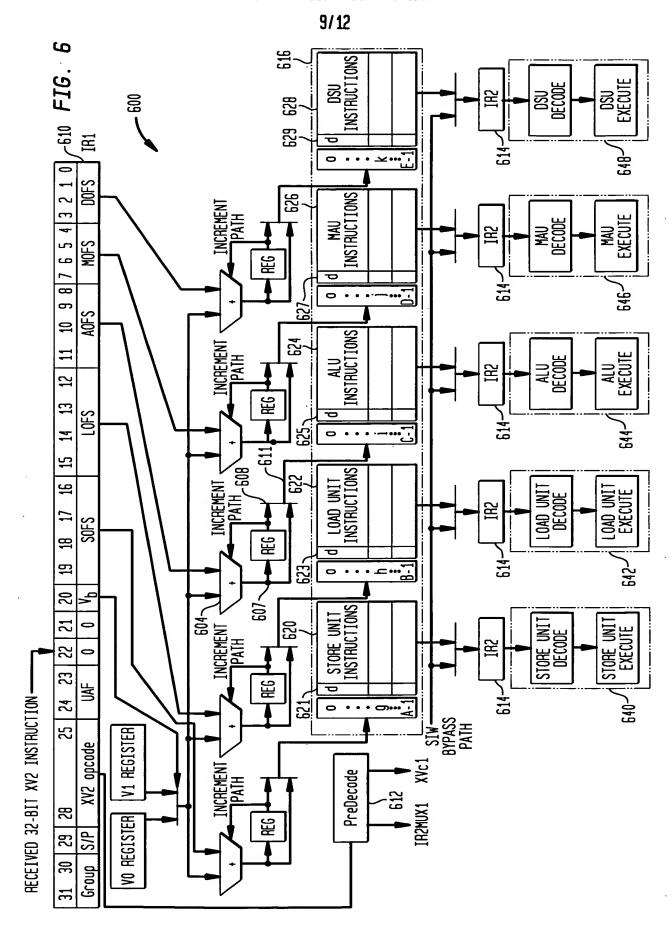
7/12

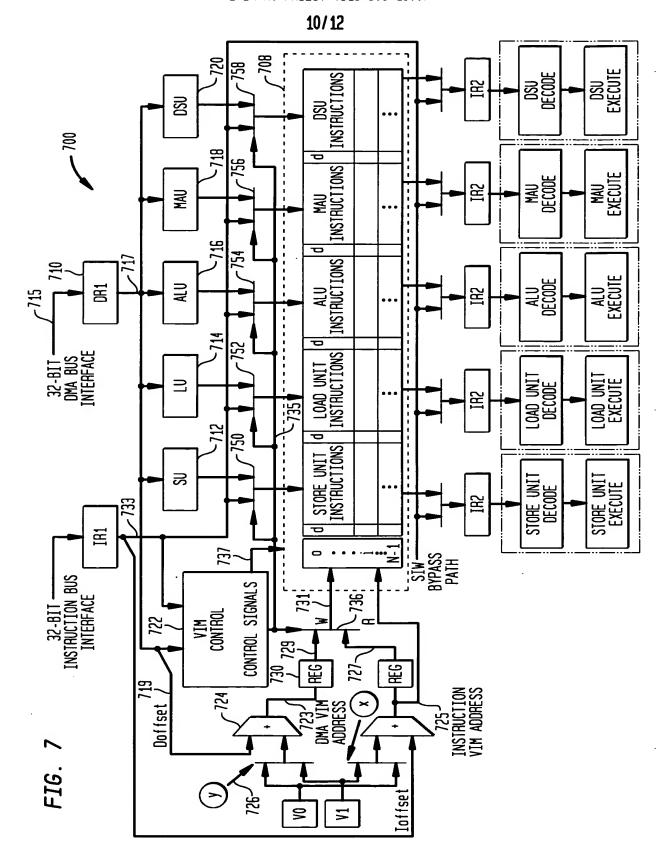
						77	F16. 4U	<u> </u>								
XV2- EXECUTE VLIW — 475	VL IW — 475									07.6						
Encoding										?~						
31 30 29	31 30 29 28 27 26 3	25 24 23	22	21	20	19	18 1,	7 16	15	14 13	12	11 10	9 8	7 6 5	25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0	
Group S/P	Ctr10p	UAF	0	0 Vb	Q.		SOFS			LOFS		AOFS	<u>ي</u>	MOFS	00FS	
	٠															
Syntax/Operation	ion						480									
Instruction		Operands	ands					<u> </u>				Operation	loi loi			_
XV2.[SP]	V[01]	, SOFFS, LOFS, AOFS, MOFS, DOFS, F=[AMDN]	OFS,	AOFS,	MOF	, i	<u> </u>		ecute ecute ecute ecute	Execute (V[01]+S0FFS)[SU] if (d=0) Execute (V[01]+L0FFS)[LU] if (d=0) Execute (V[01]+A0FFS)[ALU] if (d=0) Execute (V[01]+M0FFS)[MAU] if (d=0) Execute (V[01]+D0FFS)[DSU] if (d=0) Execute NOP @ (V[01]+unit0FS)[unit] if (d=1)	01]+	SU if [U] if ALU] i MAU] i DSU] i	if (d=0) if (d=0) if (d=0) if (d=0) Jif (d=0) JFS)[unit]	0) 0) 0) 1) 1) if (d	=	,

SERIAL NO.: 09/747.056 PETER H. PRIEST (919-806-1600)

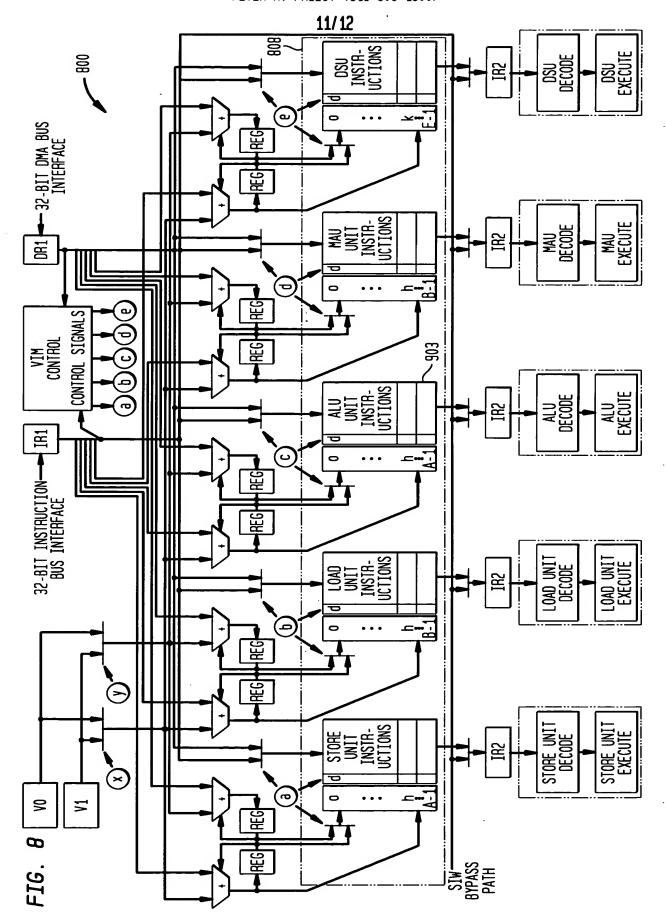


SERIAL NO.: 09/747,056 PETER H. PRIEST (919-806-1600)





SERIAL NO.: 09/747,056 PETER H. PRIEST (919-806-1600)



12/12

